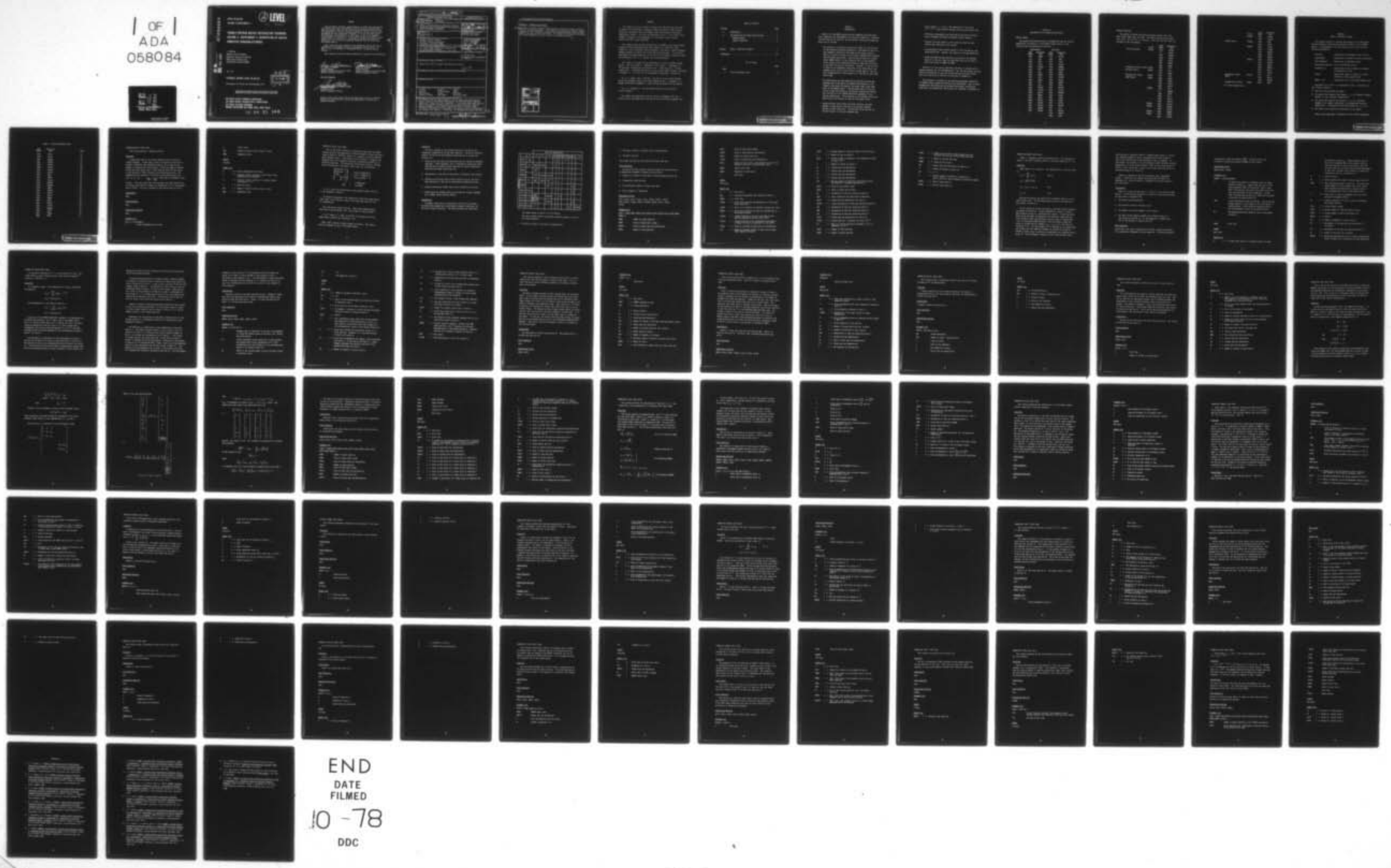
AD-A058 084 DOUGLAS AIRCRAFT CO LONG BEACH CALIF F/G 9/2

FORMAT - FORTRAN MATRIX ABSTRACTION TECHNIQUE. VOLUME II. SUPPL--ETC(U)

MAY 78 L CHAHINIAN F33615-75-C-3105

UNCLASSIFIED MDC-J7174 AFFDL-TR-66-207-VOL-2-SUPP NL

1 OF 1
ADA
058084



END
DATE
FILMED
10-78
DDC

AFFDL-TR-66-207

VOLUME II-SUPPLEMENT V

FORMAT-FORTRAN MATRIX ABSTRACTION TECHNIQUE VOLUME II, SUPPLEMENT V, DESCRIPTION OF DIGITAL COMPUTER PROGRAM-EXTENDED

L. Chahinian
Douglas Aircraft Company
McDonnell Douglas Corporation
3855 Lakewood Boulevard
Long Beach, California 90846

MAY 1978

TECHNICAL REPORT AFFDL-TR-66-207

Final Report For Period July 1975-December 1977

AIR FORCE FLIGHT DYNAMICS LABORATORY
AIR FORCE WRIGHT AERONAUTICAL LABORATORIES
AIR FORCE SYSTEMS COMMAND
WRIGHT-PATTERSON AIR FORCE BASE, OHIO 45433

[illegible]

LT. LARRY G. MOOSMAN
Project Manager
Improved Windshield Protection ADPO
Vehicle Equipment Division

[illegible]

ROBERT E. WITTMAN
Program Manager
Improved Windshield Protection ADPO
Vehicle Equipment Division

FOR THE COMMANDER:

[illegible]

SOLOMON R. METRES
Acting Director
Vehicle Equipment Division

(19) TR-66-207-VOL-2-SUPPL-5

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER (18) AFFDL-~~TR-66-207~~ VOLUME II ✓ SUPPLEMENT V	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) (6) FORMAT — FORTRAN MATRIX ABSTRACTION TECHNIQUE VOLUME II, SUPPLEMENT V, DESCRIPTION OF DIGITAL COMPUTER PROGRAM — EXTENDED.		5. TYPE OF REPORT & PERIOD COVERED (9) FINAL REPORT. July 1975 - December 1977,
		6. PERFORMING ORG. REPORT NUMBER (14) MDC-J7174
7. AUTHOR(s) (10) L. CHAHINIAN		8. CONTRACT OR GRANT NUMBER(s) (15) F33615-75-C-3105 ✓
9. PERFORMING ORGANIZATION NAME AND ADDRESS Douglas Aircraft Company ✓ McDonnell Douglas Corporation Long Beach, California 90846		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS Project: (16) 2202 Task: (17) 02 Work Unit: 01
11. CONTROLLING OFFICE NAME AND ADDRESS Air Force Flight Dynamics Laboratories (AFFDL/FEW) Air Force Wright Aeronautical Laboratories Air Force Systems Command Wright-Patterson Air Force Base, Ohio 45433		12. REPORT DATE (11) MAY 1978 (12) 85p.
		13. NUMBER OF PAGES 90
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited

63211F

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Aircraft	Dynamic	MATRIX
Analysis	Finite Element	Modal
Applications	Linear	Nonlinear
Computer Program	Math Model	Structural Model

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The FORMAT system has been augmented with new capabilities applicable to elastic stability and dynamic analyses and has been modified to improve operating efficiency and user convenience. A new large order eigen solution module and a module to evaluate matrices representing specific forms of complex exponents of the natural logarithmic base e have been added. Efficiency improvements have been made to the matrix multiply, transposition and decomposition modules. In addition, the decomposition module has been → next page

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

116400 B

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

78 08 21 108

BLOCK 20. Abstract (Continued)

cont. modified to perform congruent transformations yielding the upper triangular form of the resulting matrix. The capability to create user defined leading diagonal or full rectangular matrices with constant element values has been added to the matrix card input module. The capability of modifying available work space parameters with user input has also been provided.

FOREWORD

This report is one of a series of reports that describe work performed by Douglas Aircraft Company, McDonnell Douglas Corporation, 3855 Lakewood Boulevard, Long Beach, California 90846, under the Windshield Technology Demonstrator Program. This work was sponsored by the U. S. Air Force Flight Dynamics Laboratory, Wright-Patterson Air Force Base, under Contract F33615-75-C-3105, Project 2202/0201.

This report covers the modifications made to FORMAT, an existing computer program originally developed under USAF Contract No. F33615-71-C-1627, Project No. 1467 "Structural Analysis Methods", and Task No. 146705 "Automatic Computer Methods of Analysis for Flight Vehicle Structures". This original development was administered by the Air Force Flight Dynamics Laboratory under the cognizance of Mr. J. R. Johnson, FBR, Project Engineer.

This report is divided into two supplements to existing FORMAT documentation. They are Volume II - Supplement V and Volume V - Supplement III. The principal investigators and authors were L. Chahinian and J. Pickard, respectively. A complete description of the current FORMAT System is contained in Volumes II, V, VI, and VII, as supplemented (References 1 through 13).

Mr. D. C. Chapin, Capt., USAF Ret., was the Air Force Project Manager during the conceptual phase of the work reported herein. Lieutenant L. G. Moosman (AFFDL/FEW) succeeded Mr. Chapin during the conduct of the program.

Mr. J. H. Lawrence Jr., was the Program Director for the Douglas Aircraft Company.

This report was submitted to the Air Force on 7 December 1977, and covers the work performed during the period July 1975 through December 1977.

TABLE OF CONTENTS

LIST OF TABLES

SECTION I
INTRODUCTION

Phase II of the FØRMAT system has been augmented with additional basic capability and refinements to the existing capability. Additions and modifications to Phase II, the abstraction instruction processing phase, are as follows:

- The addition of abstraction instruction EIGENX for the solution of the eigenvalue problem statement MU = KUD wherein M and K are real symmetric matrices of the order of the static analysis. Matrix K is given in terms of its Cholesky decomposition obtained from the SEQWF abstraction. Matrix M may be input as a conventional FØRMAT matrix, or the triangular half of the symmetric triple product matrix generated through an additional variation of the SEQWF abstraction instruction. Moreover, it can be given in terms of two components consisting of the latter matrix and diagonal matrix input in the form of a column matrix. The current implementation of this program is in the form of user coded module US06.

- The SEQWF module has been additionally modified to yield the decomposition of the matrix of equation coefficients in its reverse column order followed by its normal order, both under cover of one FØRMAT matrix. The additional cost of this provision is more than offset by the savings materialized through the elimination of the FØRTRAN BACKSPACE statements in the ensuing computations. The processing of the unknown vectors now takes place in partitions as large as core space allows.

- Subroutine MATR, which reads card-input matrices, has been amended to read matrices consisting of constant elements through a single input card. The constant value could be for the full matrix, or its main diagonal only.

- Given vectors λ_r, λ_i and τ, the computation of the matrix $e^{(\lambda_r + \lambda_i)\tau}$ is now possible through the user-coded module US08.

- Efficiency improvements have been made to the matrix multiply, matrix transpose and matrix transpose multiply modules.

- Routines have been added to print execution times for most modules when operating on CDC systems.

- An experimental eigen solution module is also included as user coded module US07. However, this module is not yet operational.

- Subroutine PRØB has been modified to provide for the optional input of a SIZE card under the $RUN data which can be used to modify the values of NWØRK and KØNST.

Section II of this report summarizes the Phase II program modifications necessary for the implementation of all new and revised routines. Detailed descriptions of each new routine and each existing routine which was significantly modified are contained in Appendix A.

This document is intended for the engineer/programmer responsible for FØRMAT system program modifications and maintenance. User oriented documentation describing new and revised input/output, application techniques, and mathematical formulations are contained in Reference 15.

SECTION II

IMPLEMENTATION OF PROGRAM MODIFICATIONS

CODING CHANGES

The program modifications made to implement the new and revised capabilities into Phase II of the FØRMAT system consist of the following deck replacements and additions:

Replacements		Additions	
Deck Number	Subroutine Name	Deck Number	Subroutine Name
F000	FØRMAT	F615	WRITEU
F025	PRØB	F620	US06
F044	MATR	F621	EIGEN
F109	EUTL9	F622	VALUES
F602	SCATER	F623	CØRSØL
F604	TMPY	F624	RØRTHØ
F605	MULT	F625	CRØSS
F606	READB	F626	REDVEC
F607	TRAN	F627	REVERS
F608	SEQWF	F628	VMULTY
F609	FXTD	F629	NØRMLV
F60B	DINV	F630	READV
F60C	SRTPAK	F631	WRITEV
F60D	SPLITW	F632	STIFMX
F60E	READT	F633	MERGEM
F60F	ØRDER	F640	TSETQ
F60G	READL	F641	TIMEQ
F60H	WRITEL	F60N	FKWRIT
F60I	MATRIX	F60P	WRITEC
F60J	WRITET	F60Ø	MATSUM
F60K	DECØMP	F701	US07
F60L	ANSWER	F702	RELEIG
		F703	GENEIG
		F908	US08

OVERLAY STRUCTURE

The overlay structure given below concerns modules TMPY, MULT, TRAN, SEQWF, US06, US07, and US08. The overlay structure for decks F000, F025, F109 and F044 is unchanged. Decks F640 and F641 should be included in the root segment.

	Origin	Deck Number	Subroutine Name
Utility Routines	FMT200	F602	SCATER
		(*)	SCRACH
		F603	SQUEEZ
		F600	READA
		F601	WRITE
		F60F	ØRDER
		F60G	READL
Transpose Multiply Module	FMT220	F604	TMPY
Multiply Module	FMT220	F605	MULT
		F606	READB
Transposition Module	FMT220	F607	TRAN
SEQWF Module	FMT220	F608	SEQWF
		F60E	READT
		F60J	WRITET
		F60H	WRITEL
	FMT260	F609	FXTD
		(*)	CØMMØN
		F60A	INVTST
		F60B	DINV
		F60D	SPLITW
		F60C	SRTPAK
	FMT260	F60I	MATRIX
	FMT260	F60Ø	MATSUM
		F60P	WRITEC
	FMT260	F60K	DECØMP
	FMT260	F60N	FRWRIT
		F632	STIFMX

	Origin	Deck Number	Subroutine Name
	FMT260	F60L	ANSWER
		F615	WRITEU
EIGENX Module	FMT220	F620	US06
		F621	EIGEN
	FMT290	F622	VALUES
		F623	CØRSØL
		F624	RØRTHØ
		F625	CRØSS
		F633	MERGEM
	FMT290	F626	REDVEC
		F627	REVERS
		F628	VMULTY
		F629	NØRMLV
		F630	READV
		F631	WRITEV
Experimental Eigen Module	FMT220	F701	US07
		F702	RELEIG
		F703	GENEIG
Exponentiation Module	FMT220	F908	US08

(*) Labeled common block.

APPENDIX
PHASE II ROUTINES EXTENDED

This appendix contains a detailed description of all subroutines that were added or significantly modified in Phase II of the FØRMAT system. The documentation for each subroutine is divided into the following subheadings:

Algorithm	- description of programming logic and flow
Input/Output	- description of external data set input/output
Error Detection	- description of detectable errors
Subroutines Required	- list of subroutines called
Argument List	- description of each argument and sequence of appearance in list
Length	- approximate number (in decimal) of words required to store compiled code
Symbol List	- description of all significant symbols used

The symbol list, which is in alphanumeric order, is divided into four fields as follows:

1) The first field contains the symbol.
2) The second field contains the letters, I, L, or R denoting integer, logical, or real variable, respectively.
3) The third field contains the letters A, C, D, or U denoting argument list, common, dimensioned, or undimensioned variable, respectively. The heirachy of the above letters is A, C, D, U.
4) The fourth field contains the description of the symbol.

Table A gives page number references for each routine documented.

TABLE A. ROUTINE REFERENCE TABLE

Deck Number	Subroutine Name	Page
F109	EUTL9	10
F608	SEQWF	12
F60K	DECØMP	19
F60L	ANSWER	23
F60N	FRWRIT	28
F60Ø	MATSUM	30
F60P	WRITEC	32
F615	WRITEU	34
F620	US06	36
F621	EIGEN	43
F622	VALUES	47
F623	CØRSØL	49
F624	RØRTHØ	52
F625	CRØSS	54
F626	REDVEC	56
F627	REVERS	58
F628	VMULTY	61
F629	NØRMLV	63
F630	READV	66
F631	WRITEV	68
F632	STIFMX	70
F633	MERGEM	72
F640	TSETQ	74
F641	TIMEQ	75
F908	US08	77

PRECEDING PAGE BLANK-NOT FILMED

SUBROUTINE EUTL9 (DECK F109)

This routine expands a compressed column.

Algorithm

A compression code of zero would indicate that the column is already expanded. In that case, control is returned to the calling program. Otherwise, the compression code is set to zero and the process of expanding the column begins. Where the number of entries within the given column, NUM, is less than the order of the column, N, the locations (NUM+1) through N are initialized as zero. The compressed column consists of a series of value/location pairs as $\left(V_1, L_1, V_2, L_2, \ldots, V_{\frac{num}{2}}, L_{\frac{num}{2}}\right)$. The column is expanded in sets of pairs. Each set starts with the I-th entry and spans through the IL-th entry such that the integer row designation at IL is not greater than IL itself. The set is expanded in the reverse order of pairs.

Input/Output

None

Error Detection

None

Subroutines Required

None

Argument List

EUTL9 (A,N,NUM,KØDE)

A	Array accommodating the column

N	Column order
NUM	Number of entries within column on input
KØDE	Compression code

Length

56 words

Symbol List

A	R	A	Array accommodating the column
I	I	U	Indexing variable pointing to the first of next quantities to be relocated
IL	I	U	Indexing variable pointing to intended address of quantity at IL-1
N	I	A	Order of column
NUM	I	A	Number of entries within column on input
KØDE	I	A	Compression code

SUBROUTINE SEQWF (DECK F608)

This is the driver routine of a module which provides the FORMAT program user with the abstraction instruction SEQWF to treat the matrix equality $[ABA^T]E = [ABC^T+D]$, where B is a symmetric positive definite quasi-diagonal matrix. The solution of this equation is based on the decomposition of the symmetric congruent transformation $[ABA^T]$ by a wavefront technique. The abstraction configuration options, followed with an explanation of the matrix names used, are listed below.

$$\left\{\begin{array}{l} \text{BINV,AB,LT,E} \\ \text{BINV,AB,}\quad\text{E} \\ \text{BINV,}\quad\quad\text{E} \\ \quad\text{AB,LT,E} \\ \quad\text{AB,}\quad\text{E} \end{array}\right\} = \left\{\begin{array}{l} \text{A,B,}\pm\text{C}\cdot\text{SEQWF}\cdot\pm\text{DT} \\ \text{A,B}\quad\cdot\text{SEQWF}\cdot\pm\text{DT} \\ \text{A,B,}\pm\text{C}\cdot\text{SEQWF}\cdot \end{array}\right\}$$

$$\text{E} = \text{LT}\cdot\text{SEQWF}\cdot\pm\text{DT}$$

$$\text{ABAT} = \text{A,B}\cdot\text{SEQWF}.$$

A, B, C, and E are the matrices in the equation above, while DT is the transpose of matrix D.

By virtue of characters "INV" appearing in the first output matrix name, BINV is the inverse of matrix B and takes the place of B for all computations.

AB is the matrix product $[A]\cdot[B]$. Should the preceding matrix name contain the characters "INV", that product is then $[A]\cdot[B]^{-1}$.

LT is L^T where $L\cdot L^T = ABA^T$, or $AB^{-1}A^T$ if the name of the first output matrix contains the characters "INV".

ABAT = ABA^T, wherein B need not admit an inverse. This matrix would be intended for use in the EIGENX module.

Algorithm

The first function of this program consists of verifying the dimensional compatibility of the input matrices. Its driver functions consist of fulfilling the following conditions prior to each subroutine call:

1. Assuring that the input and output matrices are located on separate data sets. Where necessary, this is achieved by copying the input matrices onto available scratch data sets and later transferring them onto the proper output data set.

2. Positioning all data sets at the origin of the matrix they contain.

3. Optimally partitioning the available working core for the use of each subroutine in the form of conveniently dimensioned arrays.

4. Writing conventional FORMAT output matrix headers and trailers.

5. Establishing the communication with subroutines through arguments and a COMMON region labelled SCRATCH.

Input/Output

Throughput takes place in subroutines called by this program. The table below summarizes the peripheral storage of matrices, and the use of scratch data sets. The output matrices are underlined.

		DATA SET DESIGNATIONS										
		N1	N2	N3	N4	NFX	ND	NFØ	NDINVP	NXP	NLT	N5
SUBROUTINES	FXTD	$\underline{AB}$				A	B		B^{-1}			
	MATRIX	AB	$\underline{ABA^T}$	Scratch		A						
	RESEQ		ABA^T									
	RENUMB		ABA^T	Scratch	Scratch							
	MATRIX	AB	$\underline{ABC^T}$	Scratch	ABA^T	C						
	DECØMP				ABA^T						$\underline{LT}$	
	FRWRIT	Scratch									$\underline{LT}$	
	MATSUM	Scratch	$\frac{ABC^T}{ABC^T+DT}$		Scratch			DT				
	ANSWER	Scratch	$\frac{ABC^T+DT}{U^T}$		Scratch						LT	
	WRITEU	Scratch	U^T							$\underline{U}$		Scratch

The output format of matrix LT is as follows:

1. The matrix header wherein the maximum wavefront appears in lieu of the column dimension.

2. The matrix columns in the order of decomposition.

3. The matrix columns in reverse order of decomposition.

4. The matrix trailer.

This module requires not more than five scratch data sets.

Error Detection

Any one of the four reasons listed are cause for the printing of an appropriate statement followed by an error exit:

1. Inability to locate an input matrix from the specified data set.

2. Incompatible input matrices.

3. An insufficient number of scratch data sets.

4. Error flagged by a subprogram.

Subroutines Called

EUTL1, EUTL3, EUTL4, EUTL5, EUTL6, SCATER, FRWRIT, INVTST, FXTD, MATRIX, DECØMP, FRWRIT, MATSUM, ANSWER, WRITE, WRITEU, STIFMX

Argument List

SEQWF (NUMØT,NAMØT,IØSPEC,NUMIN,NAMIN,INSPEC,NUMSR,ISSPEC,NUMSC,ERRØR, NWØRK,A)

NUMØT	Number of output matrices
NAMØT	Array of output matrix names
IØSPEC	Array of output data set designations
NUMIN	Number of input matrices

NAMIN	Array of input matrix names
INSPEC	Array of input data set designations
NUMSR	Number of scratch data sets
ISSPEC	Array of scratch data set designations
NUMSC	Number of input matrix names between the equal sign and the first dot of the abstraction name
ERRØR	Logical error flag
NWØRK	Dimension of work array A
A	Work array

Length

1472 words

Symbol List

A	R	A	Work array
DXØ	L	C	Flag which designates the presence of matrix C
ERRØR	L	A	Error flag
INSPEC	I	A	Array which contains the designations of the input matrix data sets
INVERT	L	C	Matrix B is inverted on indication through this flag
IØSPEC	I	A	Array which contains the data set designations of the output matrices
ISCRCH	I	C	Integer pointing to the next available scratch data set designation within array ISSPEC
ISFXTD	I	U	Integer pointing to the alphanumeric designation of matrix AB which is contained in array NAMOT
ISSPEC	I	A	Array of available scratch data set designations
LPS	I	C	Number of columns of matrix U which have already been computed and output

LINES	I	C	Maximum number of lines per page of printed output
M	I	C	Number of rows in matrix A
MAXPF	I	C	Maximum number of elements in the compressed columns of matrix $AB^{(1)}$.
N	I	C	Number of columns in matrix A
N1	I	C	Scratch data set designation
N2	I	C	Scratch data set designation
N3	I	C	Scratch data set designation
N4	I	C	Scratch data set designation
NAC	I	C	Maximum number of interactive row/columns during the course of a Cholesky decomposition
NAMIN	I	A	Array of input matrix names
NAMØT	I	A	Array of output matrix names
ND	I	C	Designation of data set containing matrix B
NDIM	I	C	Array containing the input matrix dimensions
NDINV	I	C	Output data set designation for matrix B
NDINVP	I	C	Interim designation of data set containing matrix B
NET	I	C	Designation of data set containing matrix C
NFØ	I	C	Designation of data set containing matrix DT
NFX	I	C	Designation of data set containing matrix A
NFXTD	I	C	Output data set designation for matrix AB
NFXTDP	I	C	Interim data set designation for matrix $AB^{(-1)}$
NP	I	C	When in a double-precision environment, NP is 2; otherwise, it is 1
NUMIN	I	A	Number of input matrices
NUMØT	I	A	Number of output matrices

NUMSC	I	A	Number of input matrix names between the equal sign and the first dot of the abstraction name
NUMSR	I	A	Number of scratch data sets
NWØRK	I	A	Extent of work array A
NXP	I	A	Output data set designation for matrix ET
P	I	C	Number of columns in matrix DT
PP1	I	U	P + 1
Q	I	U	Maximum number of columns in a partition of matrix (ABC^T+DT), when processed by subroutine ANSWER
SIGNET	L	C	Sign of input matrix C
SIGNPØ	L	C	Sign of input matrix DT

SUBROUTINE DECØMP (DECK F60K)

Given A, a symmetric positive definite matrix, this routine computes L^T, an upper triangular matrix in the matrix equality $LL^T = A$.

<u>Algorithm</u>

Where matrix A is of order n, the computations of L may be stated as:

$$s_{ij} = - \sum_{k=1}^{i-1} l_{ki} l_{kj} \qquad (j>i>1)$$

$$l_{ii} = \sqrt{(a_{ii} + s_{ii})}$$

$$l_{ij} = (a_{ij} + s_{ij})/l_{ii} \qquad (j>i)$$

$$l_{ji} = 0 \qquad (j>i)$$

Data set NS contains the upper half of symmetric matrix A in reverse column order. The following takes place with the reading of each column j of matrix A.

The IEL values are stored into array B, and their column locations are stored into array LB. Should INFØ(LASTR,1) contain j, array A then contains values of s concerning this column. In that case only, these values and their corresponding row locations, taken from the first column of INFØ, are adjoined to arrays B and LB, respectively. A call to subroutine ØRDER performs a sort and merge on these two arrays to produce the JEL elements $(a_{ij} + s_{ij})$ in B, with their row locations in LB. The last element of B is replaced by its square root and divided into the remaining elements to produce the values of column L_j^T within array B, and their corresponding row locations within array LB. The off-diagonal elements of this column now make their contributions to triangular matrix S located within array A. To this end, subroutine ØRDER is used in conjunction with both columns of INFØ in the calculations of the locations of the s_{ij} elements. In the process, the contents of LB are replaced by their relative location designations, from the second column of INFØ, to avoid duplication of effort in their future use. B and LB are written onto data set NT to conclude processing of column L_j^T.

Control is returned to the calling program when j designates column 1. Otherwise, the integer variables LASTR and NEXTR are decreased by 1, and this process is repeated for the next column of the input matrix.

Input/Output

Matrix A is read from data set NS. Matrix L^T is written onto data set NLT. The following information is written for each column throughout the course of decomposition:

1. The actual column designation.

2. The diagonal values of matrices A and L.

3. The number of non-zero elements in this column.

4. The ratio of the diagonal element of A divided by that of L will cause the printing of: a) two asterisks if greater than 10^8; b) one asterisk if greater than 10^3.

Error Detection

Insufficient core space, recognized with NAC>NK, causes the printing of an appropriate statement and halts execution. Following completion of execution, should the condition $\frac{a_{ii}}{l_{ii}} > 10^8$ have arisen, the printing of an aborting message precedes an error return.

Subroutines Called

ØRDER, READT, WRITEL

Argument List

DECØMP (A,INFØ,MK,ERRØR)

A	A linear array used as a triangular matrix to store (partial) values s_{ij}. These values are kept in this array throughout their active period, that is from their first partial value until the column with which they are associated has been decomposed. At that time, the space they occupied is cleared and allocated to the next activated column of S.
INFØ	A twin-information vector of order MK. The active row/column numbers of S are stored in the first vector in ascending order. Their relative locations within A are correspondingly stored in the second vector.
MK	The maximum permissible number of active row/columns of matrix S.
ERRØR	Error flag

Length

2591 words

Symbol List

A	R	A	A linear array used as a triangular matrix to store

			the (partial) values s_{ij}. These values are kept in this array throughout their active period; that is, from the inception of their first partial until the row with which they are associated has been decomposed. At that time, the space they occupied is cleared and allocated to the next row/column of S.
M	I	A	Order of matrix L
MK	I	A	Maximum permissible number of active row/columns of S
INFØ	I	A	Twin information vector of order MK. The active row/column numbers of S are stored in the first vector. Their relative locations within triangular array A are correspondingly stored in the second vector.
B	R	D	Array which is used to accommodate the significant values of throughput matrices
LB	I	D	Integer counterpart of array B used to accommodate the column locations
LINE	I	U	Line of output counter
LINES	I	C	Maximum number of lines per page of output
LASTR	I	U	Current number of active row/columns of S
NEXTR	I	U	LASTR+1
IEL	I	U	Number of elements in arrays B and LB
NLT	I	C	Designation of the data set onto which matrix L^T is written
NS	I	C	Designation of the data set containing matrix A
NP	I	C	Number of words per real variables
IØRGNL	I	U	Row/column designations prior to optimal resequencing; LB(IEL) contains this information for each row/column

SUBROUTINE ANSWER (DECK F60L)

In the matrix expression $LL^Tx = b$, given matrices L^T and b, the former being an upper triangular matrix, this routine computes x^T by way of y, where $Ly = b$.

Algorithm

In a system of order n, the computations of each y_j column may be stated as:

$$z_{ij} = - \sum_{k=1}^{i-1} l_{ki}y_{kj} \quad (i>1)$$

$$y_{ij} = (b_{ij}+z_{ij})/l_{ii}$$

The computations of x may then be stated as:

$$v_{ij} = - \sum_{k=i+1}^{n} l_{ik}x_{kj} \quad (n>i)$$

$$x_{ij} = (y_{ij}+v_{ij})/l_{ii}$$

Costly use of the FØRTRAN BACKSPACE statement is circumvented with providing both matrices L and L^T under cover of one FØRMAT matrix. With L in reverse column order and b in reverse row order, the above recursive expressions yield the rows of x in their proper order. Furthermore, the row and column designations of L and L^T have been replaced by the allocation addresses of the corresponding columns of matrix b in array B.

The following is a description of the computations of matrix y. The next row of matrix b is read via subroutine READT in the form of the values into array A and corresponding column locations into array LA. With the provision that the columns of denominations greater than NQ will be processed through a future invocation of this routine, the appropriate values and their locations are written onto scratch data set N4 via subroutine WRITET.

For each column designation of present concern, subroutine ØRDER is invoked for allocation (new or previous) of these values within the rows of column MK of array B. Arrays A and LA are then filled with the current column of matrix L. The variable LR, which points to the corresponding column of array B, is read in the process. Where the row of b, which is now buffered in column MK of array B, corresponds to this column of L, it is added onto column LR of B, thus producing the current row of the matrix sum (b+z). Dividing this row by the last value of array A then yields the current row of matrix y.

Control is passed to the back-substitution portion of this program on one of the following conditions: a) the current column is column 1; b) the current number of active columns in matrix L is 1, and row LASDXØ of matrix b has already been read.

Otherwise, the cross-product of the newly created column of y and current column of L is deducted from array B in contribution to the formation of matrix z.

The following is a description of the computations of the final matrix, x. The current column of matrix y (y+v), which is still in column LR of array B, is divided by the current diagonal element of matrix L to produce the corresponding row of matrix x. This column is written onto N2, the output data set. Should this be column n, control is returned to the calling program. Continuing on the reading of data set NT via subroutine READL, arrays A and LA, and the associated value of LR are replenished with the next column of matrix L^T. The reading of the corresponding row of matrix y into column LR of array B is preceded and followed by backspacing data set NS. The off-diagonal elements of this column of L^T are associated with the (already computed) row of matrix x which resides in the columns of array B designated by the elements of LA. The cross-product of these associated values is deducted from the elements of y in column LR of array B to yield the corresponding row of matrix (y+v) which is now recycled in this procedure.

Input/Output

The rows of matrix b are read from data set N2. Elements of these rows which are associated with columns of denominations greater than NQ are written onto data set N4. Matrix L is read from data set NT. Matrix x is written onto data set N2.

Error Detection

None

Subroutines Called

ØRDER, WRITE, READA, READL, READT, WRITET

Argument List

ANSWER (B,LP,MK,PP1,P)

B	Storage array of dimension P by MK used to accommodate z, during the forward course of solution, and v during back-substitution
LP	A twin information vector whose first column contains the ordered LASTP column designations of B; their allocations within the columns of B are correspondingly stored in the second column of LP
MK	(MK-2) is the maximum number of active columns allowed in matrices z and v

PP1	P+1
P	Row dimension of array B

Length

2362

Symbol List

P	I	A	Number of columns in matrices x and b
PP1	I	A	P+1
MK	I	A	(MK-1) is the maximum number of active rows allowed in matrices z and v
B	R	A	Array used for the storage of matrices z and v
LASTP	I	U	The number of columns of b which have been processed during the course of computing matrix z
NEXTP	I	U	LASTP+1
LP	I	U	Twin information vector whose first column contains the ordered LASTP column designation of matrix b; their physical allocations within the columns of B are correspondingly stored in the second column of LP
A	R	D	Array which buffers the significant values of all input matrices. Moreover, it is the working array of each row of matrix L.
LA	I	D	Array which accommodates the integer values associated with array A. In the case of the rows of L, these integers represent the relative locations of the appropriate columns of array B.
IEL	I	U	Number of elements in arrays A and LA

NT	I	C	Designation of data set which contains matrix L in reverse order and matrix L^T in normal order
NS	I	C	Designation of scratch data set used to accommodate matrix y
NA	I	C	Columns of b which are of denomination greater than NQ are written onto data set N4
NQ	I	U	Each invocation of this program is to process a predetermined number of columns of array b whose denominations do not exceed NQ
LPS	I	C	LPS columns of array x have already been computed
N2	I	C	Data set containing the rows of matrix b on input and those of matrix x on output
LASTR	I	C	Number of currently active rows in matrix b
JX	I	U	During back-substitution, the next row of y is to be read when IRØW = JX
LR	I	U	The location of the currently computed row of y or x within the columns of array B
IØRGNL	I	U	Original equation sequence. Differs from IRØW where matrices L and L^T represent the optional decompositions of resequenced equations. Obtained from the last (diagonal) location of array LA.
IRØW	I	U	Row counter
LASDXØ	I	C	Known designation of last row of matrix b

SUBROUTINE FRWRIT (DECK F60N)

This routine augments a lower triangular matrix given in reverse column order with its duplicate in forward column order. This, to avoid the costly Fortran BACKSPACE commands in the ensuing multiple uses of this matrix.

Algorithm

The matrix columns are read in their reverse order via subroutine READA. In the process, should the input and output data sets differ, each column is written onto the output data set via subroutine WRITE. These columns fill array A to its capacity, each column followed by its number of elements, and compression code. Each time array A is filled to capacity its contents are written onto scratch data set N1 as one Fortran record. The forward writing of these columns begins only after column 1, the last column in the input sequence, has been read. It is then that the writing of column 2 on the output tape is followed by the others. To that end, the records that have been written on scratch data set N1 are brought back to core in the "BACKSPACE,BACKSPACE,READ" sequence. Data sets N1 and N3 are rewound as control is returned to the calling program.

Input/Output

The input matrix is read from data set N3. The output matrix is written onto data set NLT.

Error Detection

None

Subroutines Called

READA, WRITE

Argument List

FRWRIT (A)

A	Work array

Length

217 words

Symbol List

A	R	A	Work array
KØDE	I	U	FØRMAT compression code
KØL	I	U	Column designation
M	I	C	Order of matrix
N1	I	C	Scratch data set designation
N3	I	C	Input data set designation
NAC	I	C	Number of elements in the most populated matrix column
NLT	I	C	Output data set designation
NP	I	C	Number of storage words per real variable
NQ	I	U	Useful extent of array A
NUM	I	U	Number of elements in a column
NW	I	U	Pertinent number of records on scratch data set N1
NWØRD	I	C	Extent of array A
SAVE	L	U	Flag indicating a common input and output data set

SUBROUTINE MATSUM (DECK F60Ø)

This routine performs matrix summation [$\pm$ A + B] in reversed column order and compressed format. Matrix B is input in reversed column order.

Algorithm

Core is filled with partitions of sequential columns of matrix A. The algebraic sign of the elements are reversed, as required, and each column is addended with the integer representations of its number of elements and column designations. Except for the very last, each partition is written onto scratch data set N1 as a single record. The remaining task takes place in reversed column order. When the current in-core partition of matrix A has been processed, the previous partition is read from data set N1 in the BACKSPACE-READ-BACKSPACE mode. The columns of matrix B are read via subroutine READT. The values are stored into array B, their row locations are read into array LB. Merging takes place where column designations match, by adjoining the value-location pairs of B onto the indiscriminately compressed column of matrix A within array A, and sorting via subroutine ØRDER.

Input/Output

Matrix A is read from data set from data set NPØ. Matrix B is read from data set N2. The matrix sum is written onto output data set N4.

Error Detection

None

Subroutines Required

ØRDER, READA, READT, SQUEEZ, WRITE, WRITEC, WRITET

Argument List

MATSUM(A)

A	Working storage array

Length

3424 words

Symbol List

A	R	A	Work array accommodating columns of matrix A and summation columns
B	R	U	Array accommodating the value elements of columns of matrix B
DXØ	L	C	Flag designating presence of matrix B
LASDXØ	I	C	Designation of the output column of lowest denomination
LB	I	U	Array accommodating the row locations of the columns of matrix B
M	I	C	Column dimension of the matrices ·
NP	I	U	Number of storage words per real variable
NW	I	U	Counter of partitions of the A matrix
NPØ	I	C	Denomination of data set containing matrix A
N1	I	C	Scratch data set denomination
N2	I	C	Matrix B input data set denomination
N4	I	C	Output data set denomination
P	I	C	Row dimension of the matrices

SUBROUTINE WRITEC (DECK F60P)

This routine writes a compressed column in the form of all values followed by all row designations.

Algorithm

A record consisting of the column designation, the number of value/location pairs, the real elements, and their row designations is written onto data set N.

Input/Output

A column is output onto data set N.

Error Detection

None

Subroutines Required

None

Argument List

WRITEC (KØL,NUM,V,L,M,N)

KØL	Column designation
NUM	Number of (value, location) pairs
V	Array of values
L	Array of row locations
M	Row dimension of array L
N	Output data set denomination

Length

121 words

Symbol List

KØL	I	A	Column designation
NUM	I	A	Number of (value, locations) pairs
V	R	A	Array of values
L	I	A	Array of row locations
M	I	A	Row dimension of array L
N	I	A	Output data set denomination

SUBROUTINE WRITEU (DECK F615)

This routine transposes a dense matrix given in random order of rows.

Algorithm

JP is calculated as the number of columns which the available core space allows, with room to buffer the remaining extent of a row. Each row is read with its designation proper, and its leading elements are transferred to their intended locations within this partition of columns. The remaining elements, if any, are written onto data set N1 for future processing. Having thus formed this partition, each column is output onto data set NX. The N1 and N2 data set designations are interchanged at the conclusion of each partition processing.

Input/Output

The rows of the input matrix are read from data set N2. The columns are written onto data set NX.

Error Detection

None

Subroutines Called

READA, WRITE

Argument List

WRITEU (A,Q)

A	Work array
Q	Number of columns in output matrix

Length

230 words

Symbol List

A	R	A	Work array
IE	I	U	A(IE) is the last element of a buffered input row. The segments of rows starting from A(IR) are to be processed with future partitions.
IT	I	U	A(IT) is the last element within the current partition of columns
IV	I	U	A(IV) is the origin of row buffers
J	I	U	Input row designation
JL	I	U	Number of columns remaining at start of a new partition
JN	I	U	Number of columns which will have to be processed with future partitions
JP	I	U	Number of columns in current partition
JY	I	C	This matrix has (M-JY+1) non-zero rows
KØL	I	C	Output column designation
MØRE	L	U	Flag indicating future partitions
N1	I	C	Scratch data set designation
N2	I	C	Scratch data set designation
NX	I	C	Output data set designation
Q	I	A	Number of columns in output matrix

SUBROUTINE US06 (DECK F620)

This is the driver routine of the EIGENX module which provides the FØRMAT program user with the abstraction instruction USER06 to extract the dominant eigenvalues and eigenvectors represented by the expression MU = KUD. This, by way of a tridiagonal system of reduced order whose eigenvalues and eigenvectors closely approximate the desired higher end of the original eigenspectrum.

Algorithm

In the following expression M and K are symmetric matrices. M is a given positive semi-definite matrix, while the positive-definite matrix K is given in terms of its lower and upper decompositions L and L^T, where $LL^T = K$. With the matrix of eigenvectors U, and the diagonal matrix of eigenvalues D, the complete eigen-equation is:

$$MU = KUD$$

$$MU = LL^TUD$$

$$L^{-1}ML^{-T}L^TU = L^TUD$$

$$\text{Let} \qquad X = L^TU$$

$$\text{then} \qquad L^{-1}ML^{-T}X = XD$$

$$X^TL^{-1}ML^{-T}X = D$$

With interest to the leading p eigenvalues and eigenvectors only, Ojalvo and Newman (ref. 15) have demonstrated that a system of order n can be reduced to the below system of order m ($n \geq m \geq p$) wherein the desired quantities are accurately approximated:

$$\underset{(m\times n)}{X_R^T}\ \underset{(n\times n)}{\left\{L^{-1}ML^{-T}\right\}}\ \underset{(n\times m)}{X_R} = \underset{(m\times m)}{D_R}$$

with $$X_R = VY$$

Vectors V can be computed to arrive at the tridiagonal system

$$V^T L^{-1} M L^{-T} V = Y D_R Y^T$$

whose eigenvalues are found through the inspection of this Sturm sequence, hence leading to the computations of Y, X_R, and U.

The following is a derivation of the tridiagonal system:

$$V^T L^{-1} M L^{-T} V = \begin{bmatrix} \alpha_1 & \beta_1 & 0 & \cdot & & \\ \beta_1 & \alpha_2 & \beta_2 & \cdot & & \\ 0 & \beta_2 & \alpha_3 & \cdot & & \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ & & & \cdot & \alpha_{n-1} & \beta_{n-1} \\ & & & \cdot & \beta_{n-1} & \alpha_n \end{bmatrix}$$

Where v_i has just been determined

$$\begin{bmatrix} v_1^T \\ v_2^T \\ \cdot \\ \cdot \\ v_{i-2}^T \\ v_{i-1}^T \\ v_i^T \\ v_{i+1}^T \\ \cdot \\ \cdot \\ v_n^T \end{bmatrix} L^{-1}ML^{-T}v_i = \begin{bmatrix} 0 \\ 0 \\ \cdot \\ \cdot \\ 0 \\ \beta_{i-1} \\ \alpha_i \\ \beta_i \\ \cdot \\ \cdot \\ \text{Symmetry} \\ \text{(to be)} \end{bmatrix}$$

or

$$L^{-1}ML^{-T}v_i = \left[v_1 \middle| v_2 \middle| \cdots \middle| v_{i-2} \middle| v_{i-1} \middle| v_i \middle| v_{i+1} \middle| \cdots \middle| v_n \right] \begin{bmatrix} 0 \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ 0 \\ \beta_{i-1} \\ \alpha_i \\ \beta_i \\ \text{Sym-} \\ \text{metry} \\ \text{(to} \\ \text{be)} \end{bmatrix} =$$

$$v_{i-1}\,\beta_{i-1} + v_i\alpha_i + v_{i+1}\,\beta_i$$

then

$$L^{-1}ML^{-T}v_i - v_{i-1}\,\beta_{i-1} - v_i\,\alpha_i = v_{i+1}\,\beta_i = \overline{v}_{i+1}.$$

$\overline{v}_{i+1}$ is orthogonal to vectors $[v_i | v_2 \cdots\cdot | v_i] = V_K$. Proof: pre-multiplying both sides of the above equation by V_K^T

$$V_K^T\, L^{-1}ML^{-T}v_i - V_K^T\, v_{i-1}\,\beta_{i-1} - V_K^T\, v_i\,\alpha_i = V_K^T\,\overline{v}_{i+1}$$

or

$$\begin{bmatrix} 0 \\ \cdot \\ \cdot \\ \cdot \\ 0 \\ \beta_{i-1} \\ \alpha_i \end{bmatrix} - \begin{bmatrix} 0 \\ \cdot \\ \cdot \\ \cdot \\ 0 \\ \beta_{i-1} \\ 0 \end{bmatrix} - \begin{bmatrix} 0 \\ \cdot \\ \cdot \\ \cdot \\ 0 \\ 0 \\ \alpha_i \end{bmatrix} = \begin{bmatrix} 0 \\ \cdot \\ \cdot \\ \cdot \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

However, the authors of ref. (15) recommend re-orthogonalizing through the expression

$$\overline{v}_{i+1}^{\,final} = \overline{v}_{i+1} - \sum_{i=1}^{j} v_j \left\{ v_j^T \overline{v}_{i+1} \right\}$$

on the assumption that

$$v_j^T \overline{v}_{i+1} = \varepsilon.$$

Hence,

$$\overline{v}_{i+1}^{\,f} = \overline{v}_{i+1} - v_j \varepsilon = \overline{v}_{i+1} - v_j v_j^T \overline{v}_{i+1}$$

is orthogonal with v_j^T, as can be shown by premultiplying both sides

$$v_j^T \overline{v}_{i+1}^{\,f} = v_j^T \overline{v}_{i+1} - v_j^T v_j v_j^T \overline{v}_{i+1} = 0 \text{ (since } v_j^T v_j = 1).$$

The task of this program consists of partitioning work array A for the use of its subroutines. Thereafter, subroutine EIGEN is invoked to compute the eigenvalues and intermediate vectors X, through which subroutine REVERS computes the vectors U which are normalized by their components of highest absolute value in subroutine NØRMLV.

Input/Output

Matrices M and K are read from data sets N3 and NLT, respectively. Output matrix U is written onto data det NV.

Error Detection

Insufficient core space causes an error return with the printing of an appropriate statement.

Subroutines Required

EIGEN, EUTL3, EUTL4, EUTL5, EUTL6, NØRMLV, REVERS.

Argument List

EIGENX (NUMØT,NAMØT,IØSPEC,NUMIN,NAMIN,INSPEC,NUMSR,ISSPEC,NUMSC, DUMMY,ERRØR,NWØRK,A)

NUMØT	Number of output matrices
NAMØT	Array of output matrix names
IØSPEC	Array of output data set designations
NUMIN	Number of input matrices
NAMIN	Array of input matrix names
INSPEC	Array of input data set designations
NUMSR	Number of scratch data sets
ISSPEC	Array of scratch data set designations

NUMSC	Dummy argument
DUMMY	Dummy argument
ERRØR	Logical error flag
NWØRK	Dimension of work array A
A	Work array

Length

460 words

Symbol List

A	R	A	Work array
ERØR	L	C	Error flag
ERRØR	L	A	Error flag
IN	I	U	To conform to the sequence of decomposition of matrices L and L^T, the elements of matrix M are to be reordered according to a list of integers starting at A(IN)
INSPEC	I	A	Array of input data set designations
IØSPEC	I	A	Array of output data set designations
ISSPEC	I	A	Array of scratch data set designations
IU	I	U	A(IU) is the origin of a linear array of dimension N
IV	I	U	A(IV) is the origin of a linear array of dimension N
IW	I	U	A(IW) is the origin of a linear array of dimension N
LA	I	U	A(LA) is the origin of a linear array of dimension M
LB	I	U	A(LB) is the origin of a linear array of dimension M
LC	I	U	A(LC) is the origin of a linear array of dimension M
LDØF	I	U	A(LDØF) is the origin of a linear array of dimension NAC

M	I	C	The $N^{\underline{th}}$ order eigenspectrum is reduced to a tri-diagonal system of order M whose $P \leq M \leq N$ largest eigenvalues accurately represent those of the original system
N	I	C	Order of the original system
N1	I	C	Scratch data set designation
N2	I	C	Scratch data set designation
NAC	I	C	Maximum wavefront of decomposition
NAMIN	I	A	Array of input matrix names
NAMØT	I	A	Array of output matrix names
NL	I	C	Output data set designation concerning the eigenvalues
NLT	I	C	Input data set designation concerning decomposition matrices L and L^T
NMASS	I	C	Input data set designation concerning matrix M
NP	I	C	Number of machine words per real variable
NPØT	I	C	System output data set
NT	I	C	Array of scratch data set designations
NTAPE	I	C	Array of input data set designations
NUMIN	I	A	Number of input matrices
NUMØT	I	A	Number of output matrices
NUMSR	I	A	Number of scratch data sets
NV	I	C	Output data set designation concerning matrix of eigenvectors U
NWØRD	I	C	Extent of work array A
NWØRK	I	A	Extent of work array A
NX	I	C	Extent of insufficiency of work array A
P	I	C	Desired number of eigenvalues and eigenvectors

SUBROUTINE EIGEN (DECK F621)

This routine performs the computations of matrices V, D, Y, and X described in the documentation of subroutine US06 (deck F620).

Algorithm

The reverse sequence of decompositions L and L^T is read from tape NLT into array JØLD. Vector $\bar{v}_{i-1}$, which is stored in array U, is initialized as zero while vector v_i, stored in V, is initialized with random numbers generated by subroutine FLRAN. Data set NLT is positioned at the origin of matrix L^T by skipping over the records which make up matrix L. The following computations take place with each v_i:

$$\beta_{i-1}^2 = \left(\bar{v}_i^T \; v_i\right) \qquad \text{via call to function CRØSS}$$

$$\beta_{i-1} = \sqrt{\beta_{i-1}^2}$$

$$v_i = \bar{v}_i / \beta_{i-1} \qquad \text{stored on data set N1}$$

$$\left.\begin{array}{l} c_i = L^{-1} M L^{-T} v_i \\ \\ \alpha_i = v_i^T L^{-1} M L^{-T} v_i \end{array}\right\} \qquad \text{via subroutine CØRSØL}$$

$$\bar{v}_{i+1}^o = C_i - v_i \; \alpha_i - v_{i-1} \; \beta_{i-1}$$

$$\bar{v}_{i+1} = \bar{v}_{i+1}^o - \sum_{j=1}^{i} v_j \left\{ v_j^T \; \bar{v}_{i+1}^o \right\} \qquad \text{via subroutine RØRTHØ}$$

In this process, the values of α, β^2 and β are stored in arrays A, S and B respectively, and the variable T is computed as the maximum of sums $(\beta_{i-1} + \alpha_i + \beta_i)$.

The eigenvalues of the symmetric tridiagonal matrix (whose elements are now described with the contents of array A for the diagonal values, and array S for the super and sub-diagonal values) are extracted in subroutine VALUES and stored in array W which was previously initialized with the value of T as upper bound. Subroutine REDVEC then computes the vectors Y, and subroutine VMULTY computes the X vectors.

Input/Output

Data set NLT is positioned at the origin of matrix L^T. Data set NMASS is positioned at the origin of matrix M. The P eigenvalues from array W are written onto output data set NL.

Error Detection

The inability to locate matrix M from data set NMASS or a dimensional disagreement between matrices M and L will cause an error return with the printing of an appropriate statement.

Subroutines Required

CØRSØL, CRØSS, EUTL3, EUTL5, EUTL6, FLRAN, READA, REDVEC, RØRTHØ, VALUES, VMULTY, WRITE

Argument List

EIGEN (C,U,V,W,A,B,S,LDØF,JØLD,NAMIN,N)

C	Array used to accommodate vector c_i
U	Array used to accommodate vector v_i

V	Array used to accommodate vectors $\overline{v}_{i+1}^{(1)}$ and $\overline{v}_{i+1}^{final}$
W	Array used to accommodate vector $\overline{v}_{i+1}^{(1)}$ and the eigenvalues
A	Array of α's
B	Array of β's
S	Array of β^2's
LDØF	Array used by subroutine CØRSØL
JØLD	Array accommodating the re-ordered sequence of decompositions L and L^T
NAMIN	Array of input matrix names
N	Order of input matrices

Length

494 words

Symbol List

A	R	A	Array of α's
ALPHA	R	U	α_i
B	R	A	Array of β's
BETA	R	U	β_i
BETASQ	R	U	β_i^2
C	R	A	Array used to accommodate vector c_i
ERRØR	L	C	Error flag
JØLD	I	A	Array accommodating the re-ordered sequence of decompositions L and L^T
M	I	C	Order of tridiagonal system
N	I	A	Order of eigenspectrum

N1	I	C	Scratch data set designation used to accommodate the v vectors
NAMIN	I	A	Array of input matrix names
NL	I	C	Output data set designation concerning the array of eigenvalues
NLT	I	C	Designation of data set containing matrices L and L^T
NMASS	I	C	Designation of data set containing matrix M
LDØF	I	A	Array used by subroutine CØRSØL
NPØT	I	C	System output data set
NX	I	C	Vector counter
P	I	C	Number of requested eigenvalues and corresponding eigenvectors
S	R	A	Array of β^2's
T	R	U	Largest modulus of a column of the tridiagonal system
TN	R	U	Modulus of a column of the tridiagonal system
U	R	A	Array accommodating a vector v_i
V	R	A	Array accommodating a vector $\bar{v}_{i+1}^{(1)}$ and $\bar{v}_{i+1}^{final}$
W	R	A	Array accommodating a vector $\bar{v}_{i+1}^{(1)}$ and the eigenvalues

SUBROUTINE VALUES (DECK F622)

This routine determines the eigenvalues of a tridiagonal system through inspection of the Sturm sequence.

Algorithm

The array of eigenvalues, V, has been initialized with the largest norm of the tridiagonal system prior to entry into this routine. The variable C, which is to reflect the lower bound of the lowest eigenvalue being sought, is initialized with the negative value of the largest norm. The eigenvalues are determined in the reverse order of algebraic magnitude. The reverse counter NZ is set to the order of the system, NX. The attempted eigenvalue, U, is set to (V(NZ) + C)/2. The determinant of the system (A-UI) is computed as the variable NC counts the sign changes between successive principal minors. Should this variable reach the value of NZ, U is smaller than the NZth eigenvalue; C is then set to U, and a new attempt is made. Otherwise, the values of V from NC + 1 through NZ which are smaller than U are set to U. This process is repeated for the next algebraically larger eigenvalue. Having thus determined all NX eigenvalues, these are shifted as necessary to place the P largest in absolute value in leading position.

Input/Output

None

Error Detection

None

Subroutines Required

None

Argument List

VALUES (A,B,V)

A	Main diagonal of tridiagonal matrix
B	Super/sub-diagonal of tridiagonal matrix
V	Array of eigenvalues on exit from this routine

Length

250 words

Symbol List

A	R	A	Main diagonal of tridiagonal system
b	R	A	Super/sub-diagonal of tridigonal system
C	R	U	Lower bound of lowest eigenvalue
P	I	C	Required number of eigenvalues largest in magnitude
PØ	R	U	Previous leading minor of tridiagonal system
P1	R	U	Current leading minor of tridiagonal system
PM	R	U	PØ after computation of P1
SIGNPØ	L	U	PØ is positive when SIGNPØ is true
SIGNP1	L	U	P1 is positive when SIGNP1 is true
NC	I	U	Sign change counter between successive principal minors
NX	I	C	Order of tridiagonal system
NZ	R	U	Eigenvalue counter
U	R	U	Attempted eigenvalue
V	R	A	Exit array of eigenvalues

SUBROUTINE CØRSØL (DECK F623)

Given vector v, the symmetric positive semi-definite matrix M, and triangular matrices L and L^T, where $LL^T = K$, as K is a symmetric positive-definite matrix, this routine computes the scalar product $v^T L^{-1} M L^{-T} v$ and the vector $L^{-1} M L^{-T} v$.

Algorithm

The back-substitution operations concerning the expression $L^T b = v$ yield $b = L^{-T} v$. As generated by subroutine DECØMP of FØRMAT's SEQWF module, matrices L and L^T are input under cover of one matrix. Thus, the elements of vector b are computed and stored in the reordered sequence. The expression $d = Mb = ML^{-T} v$ is computed by reading and cross-multiplying each (transposed) column of M with vector b. A distinction is made here between a conventional FØRMAT M matrix, and the M matrix that was generated in its symmetric half by the SEQWF module. The latter would be the case where the logical variable SEQWFM is true. The rows and columns of matrix M comply to the sequence of b through information vector JØLD. That is M_{ij} is resubscripted M_{kl} where K = JØLD(I) and L = JØLD(J). Since $v^T L^{-i} M L^{-T} v = L^{-T} v\ {}^T M L^{-T} v = b^T d$, each computed element of d is multiplied by the corresponding element of b to contribute to the scalar product. At this time control is returned to the calling program where this routine has been invoked for the last time. Otherwise, the forward elimination operations concerning the expression $Lc = d$ yield c in place of d. Note that $c = L^{-1} d = L^{-1} M L^{-T} v$, the desired vector.

Input/Output

Matrices L^T and L are read from data set NLT. Matrix M is read from data set NMASS.

Error Detection

None

Subroutines Required

READA, READL

Argument List

CØRSØL (V,D,JØLD,LDØF,ALPHA,NAMIN)

V	Array of dimension N containing vector v on entry and vector b on exit
D	Array of dimension N accommodating intermediate vector d and output vector c
JØLD	The elements of matrix M are ordered according to this array. That is, M_{ji} is transformed to M_{kl} as K = JØLD(I), and L = JØLD(J)
LDØF	Array whose elements represent the relative locations of the elements of vectors v and d
ALPHA	Variable representing the scalar quantity $v^TL^{-1}ML^{-T}v$
NAMIN	Array containing the FØRMAT name of matrix L (and L^T)

Length

1898 words

Symbol List

A	R	U	Storage array for the columns of matrix M and the real values of the columns of matrices L and L^T
ALPHA	R	A	Variable representing the scalar quantity $v^TL^{-1}ML^{-T}v$
D	R	A	Array of dimension N which accommodates vectors d and c
IEL	I	U	Number of value-location pairs in a column of L or L^T

KØL	I	U	Matrix M column designations
LA	I	U	Array accommodating the integers corresponding to the values of array A
LR	I	U	Variable read with each column of L and L^T pertaining to the corresponding elements of vectors v, b and d
M	I	C	Number of times this routine is to be executed
N	I	C	Order of matrices
NAC	I	C	Maximum wavefront
NAMIN	I	A	Array containing the FØRMAT name of matrix L (and L^T)
NEL	I	U	IEL-1
NLT	I	C	Designation of the data set containing matrices L and L^T under cover of one FØRMAT matrix
NMASS	I	C	Designation of data set containing matrix M
NX	I	C	Number of times this routine has been entered
V	R	A	Array of dimension N containing vector v on entry and vector b on exit
SEQWFM	L	C	The M matrix is the triangular half of the symmetric triple product matrix generated by the SEQWF module when SEQWFM is true

SUBROUTINE RØRTHØ (DECK F624)

This routine reorthogonalizes a newly computed eigenvector with previously computed mutually orthogonal eigenvectors.

Algorithm

As described in the documentation of subroutine EIGEN X, the process of reorthogonalizing vector v_i^o with previously computed mutually orthogonal vectors v_j, $(1 \leq j < i)$, may be executed through the expression $v_i = v_i^o - \sum_{j=1}^{i-1} v_j \left\{ v_j^T v_i^o \right\}$.

Arrays U and V contain v_i^o. The latter array will ultimately contain the desired reorthogonalized vector. Each vector v_j read into array C from data set N1 is transposed cross-multiplied with v_i^o in array U; that product is then premultiplied with v_j, the result is deducted from the contents of array V and stored into array V.

Input/Output

Vectors v_j are read from data set N1.

Error Detection

None

Subroutines Required

CRØSS

Argument List

RØRTHØ (U,V,C,N)

U	Array containing vector v_i^o
V	Array containing vector v_i^o on input, and v_i on exit

C	Array used for the storage of vectors v_j
N	Order of vectors

Length

130 words

Symbol List

C	R	A	Array used for the storage of vectors v_j
E	R	U	$v_j^T v_i^o$
N	I	A	Order of vectors
U	R	A	Array containing vector v_i^o
V	R	A	Array containing vector v_i^o on input and v_i on exit
N1	I	C	Designation of data set containing vectors v_j
NX	I	C	Value of subscript 'i'

FUNCTION CRØSS (DECK F625)

This function subprogram computes the cross-product of two given vectors.

Algorithm

This function is computed as the cross-product of given vectors A and B.

Input/Output

None

Error Detection

None

Subroutines Required

None

Argument List

CRØSS (A,B)

A	Given row vector
B	Given column vector

Length

82 words

Symbol List

A	R	A	Given row vector
B	R	A	Given column vector

I	I	U	Indexing variable
N	I	C	Order of vectors A and B

SUBROUTINE REDVEC (DECK F626)

This routine performs the complete eigensolution of a real symmetric tridiagonal system using the method of Jacobi. Computation is restricted to the upper half of the matrix.

Algorithm

Array A is initialized to reflect the contents of array D as its diagonal and those of array U as its super-diagonal. Array X, the ultimate orthogonal matrix of eigenvectors, is initialized as an identity matrix. The reader is referred to reference (16) for a comprehensive derivation of this method. The annihilation of off-diagonal elements progresses from one column to the next, following a complete new sweep across the columns previously processed. An element is annihilated only on condition that it be larger in magnitude than the largest previously computed zero. The computations of the matrix of eigenvectors take place concurrently.

Input/Output

None

Error Detection

None

Subroutines Required

None

Argument List

REDVEC (X,A,D,U,M)

X	Array for eigenvectors

A	Array accommodating the tridiagonal matrix to be diagonalized
D	Array accommodating the diagonal elements of the matrix to be diagonalized
U	Array accommodating the superdiagonal of the matrix to be diagonalized
M	Order of the eigen-equation

Length

540 words

Symbol List

A	R	A	Array accommodating the matrix to be diagonalized
AGAIN	L	U	Flag calling for the continuation of the diagonalization process
BIG	R	U	Value of largest computed zero
D	R	A	Array accommodating the diagonal elements of the matrix to be diagonalized
M	I	A	Order of the eigen-equation
U	R	A	Array accommodating the superdiagonal of the matrix to be diagonalized
X	R	A	Array of eigenvectors on exit from this routine

SUBROUTINE REVERS (DECK F627)

This routine performs the matrix back-substitution $L^T X = V$ where neither matrix fits core.

Algorithm

Matrix L^T was generated by the FORMAT SEQWF module's subroutine DECØMP. Recall the computations of each column X_j:

$$y_{ij} = - \sum_{k=i+1}^{n} l_{ik} \, x_{kj} \qquad (n > i)$$

$$x_{ij} = (v_{ij} + y_{ij})/l_{ii}$$

Each column of L^T is read from data set NLT with the variable LR designating the column of array V which is allocated to the corresponding row of matrix v. That row is then read into the array column from data set N3 via subroutine READV. During the back-substitution operations, the correspondence between the rows of L^T and the columns of array V is achieved through the elements of array LA which were read as part of the column of L^T. Thus processing a column of L^T yields the corresponding row of x. The original designation of that row, taken from the element of array LA, is written onto data set N2 with that row.

Input/Output

Matrix L^T is read from data set NLT. Matrix V is read from data set N3. The rows of matrix X are written onto scratch data set N2.

Error Detection

None

Subroutines Required

READL, READV, WRITE

Argument List

REVERS (V,P)

V	Array
P	Column dimension of matrices V, Y and X

Length

1710 words

Symbol List

A	R	U	Array accommodating the values of columns of matrix L^T
D	R	U	Diagonal values of L^T
IEL	I	U	Number of elements in a column of L^T
LA	I	U	Array accommodating the correspondence between the row elements of L^T stored in array A and the columns of array V
LR	I	U	Designation of the column of array V corresponding to the current column of L^T
N	I	C	Order of matrix L^T
N2	I	C	Scratch data set onto which the rows of matrix X are written
IEL	I	U	Number of elements in a column of L^T
NEL	I	U	IEL-1
NLT	I	U	Data set containing the columns of L^T
IØRGNL	I	U	Original designation of current equation

P	I	A	Column dimension of matrices V, U and X
V	I	A	Array whose columns accommodate rows of matrices V, Y and X

SUBROUTINE VMULTY (DECK F628)

This routine performs the matrix product $X^T = V^TY$, wherein Y is core resident.

Algorithm

The integer variable IN is calculated as the number of rows of X which array A can accommodate with the unused extent of a column of V starting from location IV. The computations of each partition of X are effected with the summations of the partial products of each column of V and the corresponding row of Y. In the process, elements of V which do not enter into the computations of the current partition are written onto data set N2 via a call to subroutine WRITEV. The computed rows are then written onto data set N3. The N1 and N2 data sets are rewound, and their designations are interchanged at the conclusion of processing each partition.

Input/Output

Matrix V is read from data set N1. The product matrix is output on data set N3.

Error Detection

None

Subroutines Required

READV, WRITEV

Argument List

VMULTY (Y,A,M)

Y	Array accommodating matrix Y

A	Work array
M	Row dimension of Y

Length

266 words

Symbol List

A	R	A	Work array
IN	I	U	Number of rows in a partition of X
IT	I	U	PxIN
IV	I	U	Origin of the columns of V within array A
LM	I	U	The segments of the columns of V starting from A(LM) concern future partitions of X
M	I	A	Row dimension of matrices V and Y
MØRE	L	U	Flag designating future partitions of X
N	I	C	Column dimension of matrix V
NL	I	U	Current extent of the columns of V
NM	I	U	Extent of the columns of V for the computations of the next partition of X
NØWRD	I	C	Dimension of array A
N1	I	C	Designation of the data set which contains the columns of V
N2	I	C	Designation of the data set onto which are written the extents of columns of V concerning the computations of future partitions of X
N3	I	C	Output data set designation
P	I	C	Column dimension of matrix Y
X	R	U	Variable representing elements of V

SUBROUTINE NØRMLV (DECK F629)

This routine normalizes row-input eigenvectors on their largest absolute components and outputs them by column.

Algorithm

Having computed the number of column vectors that core space allows, each row is read from data set N2 and the elements beyond the last accommodable are written onto data set N1 for future processing. Each column thus retained in core is inspected for its largest absolute component and divided by that value. It is then output onto data set NV via subroutine WRITE. At the conclusion of processing each partition, the N1 and N2 data set designations are interchanged and the process is repeated until all columns have been processed.

Input/Output

The rows of the input matrix are read from data set N2. Data set N1 is used as a scratch data set. The final columns are output onto data set NV.

Error Detection

None

Subroutine Required

READA, WRITE

Argument List

NØRMLV (A)

A	Work array

Deck Length

252

Symbol List

A	R	A	Work array
B	R	U	Normalizing value of each vector
IE	I	U	A(IE) is the last element of each buffered row which is to be part of the current partition of vector columns
IR	I	U	A(IR) is the first element of each buffered row which is to be part of future partitions
IT	I	U	Number of words in the current partition of vector columns
IV	I	U	A(IV) is the origin of row buffer
JD	I	U	Vector column number
JL	I	U	Number of vector columns yet to be processed
JN	I	U	Number of column vectors in future partitions
JP	I	U	Number of column vectors in current partition
KE	I	U	A(KE) is the last element of a column vector
KS	I	U	A(KS) is the origin of a column vector
MØRE	L	U	Flag denoting future partitions
N	I	C	Order of column vectors
NV	I	C	Output data set designations
NWØRD	I	C	Extent of work array A
N1	I	C	Rows making up future partitions of vectors are written onto this data set

N2	I	C	The vector rows are read from this data set
P	I	C	Number of vector columns

SUBROUTINE READV (DECK F630)

This routine reads a dimensioned linear array from a specified data set.

Algorithm

Array A, of dimension N, is read from data set M as control is returned to the calling program.

Input/Output

Array A is read from data set M.

Error Detection

None

Subroutines Required

None

Argument List

READV (A,N,M)

A	Array of dimension N
N	Dimension of array A
M	Input data set designation

Length

72 words

Symbol List

A	R	A	Array of dimension N

N	I	A	Dimension of array A
M	I	A	Input data set designation

SUBROUTINE WRITEV (DECK F631)

This routine writes a dimensioned array onto a specified data set.

Algorithm

Array A, of dimension N, is written onto data set M as control is returned to the calling program.

Input/Output

Array A is written onto data set M.

Error Detection

None

Subroutines Required

None

Argument List

WRITEV (A,N,M)

A	Array of dimension N
N	Dimension of array A
M	Output data set designation

Length

72 words

Symbol List

A	R	A	Array of dimension N

N	I	A	Dimension of array A
M	I	A	Output data set designation

SUBROUTINE STIFMX (DECK F632)

This routine transcribes a matrix of triangular form, written in reverse order, from a temporary device to the output data set. This matrix does not adhere to the FØRMAT convention, but can be copied, as through the RENAME abstraction. It is intended strictly for subsequent use in the EIGENX module.

Algorithm

The last value/location pair of each record is dropped prior to its being written onto the output data set. The column designation within the matrix header is set negative to distinguish this special matrix.

Input/Output

None

Error Detection

None

Subroutines Required

EUTL5, EUTL6, READT, WRITEL

Argument List

STIFMX (NAMØT,IØSPEC,A,LA,IEL)

NAMØT	FØRMAT matrix name
IØSPEC	Output data set designation
A	Array accommodating the real words
LA	Integer counterpart of A

IEL	Dimension of A and LA

Length

164 words

Symbol List

A	R	Array used to buffer real words
IEL	I	Dimension of A and LA
IØSPEC	I	Output data set designation
LA	I	Array used to buffer integers
NAMØT	I	FØRMAT matrix name

SUBROUTINE MERGEM (DECK F633)

This routine performs the summation of a sparse symmetric matrix, represented in its upper half only, with a diagonal matrix input as a column matrix algorithm.

Algorithm

The symmetric matrix was generated by FORMAT's SEQWF module. It is recognized through the artifice of a negative value in place of the column dimension in the matrix header. The value itself is the denomination of the lowest (last) existing column. The summation takes place with the diagonal matrix in permanent core residence and the symmetric matrix read a column at a time.

Input/Output

The diagonal matrix is read as a column matrix from data set NMD. The upper half of the symmetric matrix is read from data set NMASS. The sum of these matrices is written onto data set N1.

Error Detection

The inability to locate an input matrix from its designated data set, dimensional incompatibilities or failure of the symmetric matrix to be SEQWF module generated are cause for error return with the printing of an appropriate statement.

Subroutines Required

EUTL1, EUTL3, EUTL5, EUTL6, READA, READL, WRITEL

Argument List

MERGEM (A,NAMIN)

A	Work array

NAMIN Array of input matrix names

Length

410 words

Symbol List

A	R	A	Work array
LAST	I	U	Lowest row location of the symmetric matrix
MØRED	L	U	When "true" usage of the diagonal matrix has not been completed
MØREM	L	U	When "true" usage of the symmetric matrix has not been completed
NAMIN	I	A	Array of the input matrix names
NPØT	I	C	Systems' output data set
NP	I	C	NP is 1 for single precision, and 2 for double precision
READM	L	U	When "true" the current column denomination of the symmetric matrix has not yet been read
SEQWFM	L	C	When "true", the symmetric matrix is a SEQWF module generated matrix as expected

SUBROUTINE TSETQ (DECK 640)

This routine initializes the CPU time clock.

Algorithm

The call to subroutine TIMREM provides the CPU seconds remaining for the execution of this job. That value is stored for future reference in the variable ØRGTIM located within labelled common CLØCK.

Input/Output

None

Error Detection

None

Subroutines Required

TIMREM

Argument List

None

Length

7 words

Symbol List

ØRGTIM	R	C	Execution time remaining

SUBROUTINE TIMEQ (DECK F641)

This routine measures the CPU time elapsed since subroutine TSETQ was last invoked.

Algorithm

The CPU time remaining for the execution of this job is provided through a call to subroutine TIMREM. That quantity is substracted from the variable ØRGTIM in labelled common block CLØCK, which contains the time remaining when subroutine TSETQ was last invoked, to yield the appropriate elapsed time.

Input/Output

None

Error Detection

None

Subroutines Required

TIMREM

Argument List

TIMEQ (TCPU,TIØ)

TCPU	On exit from this routine, TCPU contains the CPU seconds elapsed since subroutine TSETQ was last invoked
TIØ	Not used at this time

Length

14 words

Symbol List

ØRGTIM	R	C	Execution time remaining
TCPU	R	A	CPU seconds elapsed since subroutine TSETQ was last invoked
TIØ	R	A	Not used

SUBROUTINE US08 (DECK F908)

Given vectors λ_r, λ_i and τ, this routine generates the matrix $F(\tau) = e^{(\lambda_r + i\lambda_i)\tau}$.

Algorithm

$F(\tau) = e^{(\lambda_r + i\lambda_i)\tau} = e^{\lambda_r\tau} [\cos(\lambda_i\tau) + i\sin(\lambda_i\tau)]$. The absence of λ_i is understood to be the case with only 2 input vectors. In that case $F(\tau) = e^{\lambda_r\tau}$. The general (complex) problem has 2 output matrices, the first being the real component, and the second the imaginary component. A column of output is computed for each τ element.

Input/Output

The input matrices are read from data sets designated by the last column of array INSPEC. The output matrices are written onto data sets designated by the last column of array IØSPEC.

Error Detection

Failure to locate an input matrix is cause for error return with the printing of an appropriate message.

Subroutines Required

EUTL3, EUTL5, READA, SQUEEZ

Argument List

US08 (NUMØT,NAMØT,IØSPEC,NUMIN,NAMIN,INSPEC,NUMSR,ISSPEC,NUMSC,DUMMY, ERRØR,NWØRK,A,IPRINT)

NUMØT	Number of output matrices in this FØRMAT abstraction
NAMØT	Array containing the alphanumeric characters making up the output matrix name

IØSPEC	Array which contains the designations of the output matrix data sets
NUMIN	Number of input matrices
NAMIN	Array whose columns contain the alphanumeric characters making up the input matrix names
INSPEC	Array which contains the designations of the input matrix data sets
NUMSR	Number of available scratch data sets
ISSPEC	Array containing the designations of available scratch data sets
NUMSC	Dummy argument
SCALAR	Dummy argument
ERRØR	Logical error flag
NWØRK	Extent of work array A
A	Work array
IPRINT	Dummy argument

<u>Length</u>

625 words

<u>Symbol List</u>

JS	I	U	Origin of τ within array A
JE	I	U	Ending of τ within array A
LAMIS	I	U	Origin of λ within array A
LAMIE	I	U	Ending of λ within array A

REFERENCES

1. J. P. Cogan, Jr., FORMAT II-Second Version of Fortran Matrix Abstraction Technique; Volume II, Description of Digital Computer Program, AFFDL-TR-66-207, Volume II, Air Force Flight Dynamics Laboratory, Wright-Patterson Air Force Base, Ohio, March 1967.

2. W. J. Lackey, R. E. Wild, FORMAT II-Second Version of Fortran Matrix Abstraction Technique; Volume II, Supplement I. Description of Digital Computer Program, AFFDL-TR-66-207, Volume II, Supplement I, Air Force Flight Dynamics Laboratory, Wright-Patterson AFB, Ohio, November 1968.

3. C. G. Hooks, FORMAT II-Second Version of Fortran Matrix Abstraction Technique; Volume II, Supplement II. Description of Digital Computer Program System/360, AFFDL-TR-66-207, Volume II, Supplement II, Air Force Flight Dynamics Laboratory, Wright-Patterson AFB, Ohio, February 1969.

4. W. J. Lackey, S. H. Miyawaki, FORMAT - Fortran Matrix Abstraction Technique; Volume II, Supplement III. Description of Digital Computer Program - Extended, AFFDL-TR-66-207, Volume II, Supplement III, Air Force Flight Dynamics Laboratory, Wright-Patterson Air Force Base, Ohio, June 1970.

5. L. Chahinian, S. H. Miyawaki, FORMAT - Fortran Matrix Abstraction Technique; Volume II, Supplement IV. Description of Digital Computer Program - Extended, AFFDL-TR-66-207, Volume II, Supplement IV, Air Force Flight Dynamics Laboratory, Wright-Patterson AFB, Ohio, April 1973.

6. J. Pickard, FORMAT - Fortran Matrix Abstraction Technique; Volume V. Engineering User and Technical Report, AFFDL-TR-66-207, Volume V, Air Force Flight Dynamics Laboratory, Wright-Patterson AFB, Ohio, October 1968.

7. J. Pickard, FORMAT - Fortran Matrix Abstraction Technique; Volume V, Supplement I. Engineering User and Technical Report - Extended, AFFDL-TR-66-207, Volume V, Supplement I, Air Force Flight Dynamics Laboratory, Wright-Patterson AFB, Ohio, June 1970.

8. J. Pickard, FORMAT - Fortran Matrix Abstraction Technique; Volume V, Supplement II. Engineering User and Technical Report - Extended, AFFDL-TR-66-207, Volume V, Supplement II, Air Force Flight Dynamics Laboratory, Wright-Patterson AFB, Ohio, April 1973.

9. J. P. Cogan, Jr., R. C. Morris, and J. R. Wells, FORMAT - Fortran Matrix Abstraction Technique; Volume VI. Description of Digital Computer Program - Phase I, AFFDL-TR-66-207, Volume VI, Air Force Flight Dynamics Laboratory, Wright-Patterson AFB, Ohio, September 1968.

10. R. C. Morris, FORMAT - Fortran Matrix Abstraction Technique; Volume VI, Supplement I. Description of Digital Computer Program - Phase I - Extended, AFFDL-TR-66-207, Volume VI, Supplement I, Air Force Flight Dynamics Laboratory, Wright-Patterson AFB, Ohio, June 1970.

11. R. C. Morris, FORMAT - Fortran Matrix Abstraction Technique; Volume VI, Supplement VI, Supplement II. Description of Digital Computer Program - Phase I - Extended, AFFDL-TR-66-207, Volume VI, Supplement II, Air Force Flight Dynamics Laboratory, Wright-Patterson AFB, Ohio, April 1973.

12. R. C. Morris, J. R. Wells, and P. S. Yoon, FORMAT - Fortran Matrix Abstraction Technique; Volume VII. Description of Digital Computer Program - Phase III, AFFDL-TR-66-207, Volume VII, Air Force Flight Dynamics Laboratory, Wright-Patterson AFB, Ohio, September 1968.

13. J. A. Frank, FORMAT - Fortran Matrix Abstraction Technique; Volume VII, Supplement I. Description of Digital Computer Program - Phase III - Extended, AFFDL-TR-66-207, Volume VII, Supplement I, Air Force Flight Dynamics Laboratory, Wright-Patterson AFB, Ohio, June 1970.

14. R. J. Melosh and R. M. Bamfort, "Efficient Solution of Load-Deflection Equations," Journal of the Structural Division, ASCE, Volume 95, No. ST 4, April 1969, pp 661-676.

15. I. U. Ojalvo and M. Newman, "Vibration Modes of Large Structures by an Automatic Matrix Reduction Method," AIAA Journal, July 1970, pp 1234-1239.

16. J. Pickard, FORMAT - Fortran Matrix Abstraction Technique; Volume V, Supplement III. Engineering User and Technical Report - Extended, AFFDL-TR-66-207, Volume V, Supplement III, Air Force Flight Dynamics Laboratory, Wright-Patterson AFB, Ohio, MAY 1978.